Reporting of Task 2A

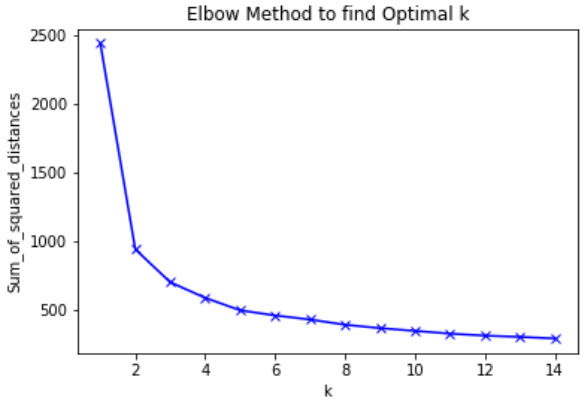
```
Accuracy of decision tree: 72.131%
Accuracy of k-nn (k=5): 81.967%
Accuracy of k-nn (k=10): 86.885%
```

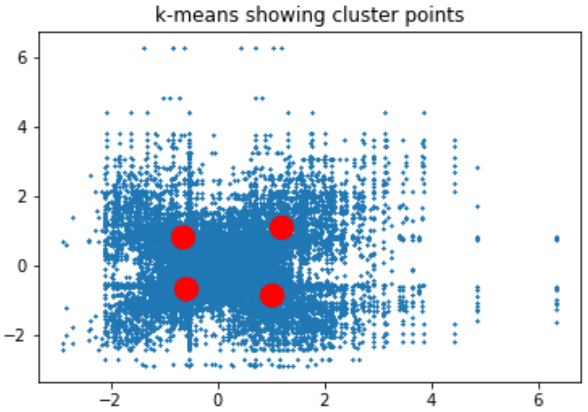
K-NN algorithm performed better than the decision tree on this dataset. This is because overfitting can occur in decision trees, even though a maximum depth is set, there could be many samples in each leaf which can lead to incorrect predictions when compared to the truth values and a decision tree could be too simple for a data with many dimensions. For k-nn, k = 10 performed better because k = 5, is more sensitive to noise points, k = 10 is not too small or large, so the chances of points included from other neighbours are low.

Before we can apply data to classification algorithms, the data in world.csv is compared to life.csv. This is to remove data in world.csv that does not have a corresponding life expectancy at birth (y) value, so that the classifier can be trained with an existing y value. Also, world.csv has non-numerical values ("..") so these are considered missing values and are imputed by using the median value of each feature. Afterwards, the data is split between the training set (the set we use to fit our classifiers with the corresponding y training values) and the testing set, the set we use to predict our y values. This is because if we test on the training set directly then we would get overfitting, it would not be a pair performance test. Furthermore, the data is treated to standardisation because k-nn makes use of Euclidean distance (the nearness between points), and the majority vote based on the k nearest points. If it was not standardised, the distances will not be informative. The mean, variance and median of each feature is found using scaler.mean_, scaler.var_ and np.median and is output to task2a.csv. The data is then used to fit the classifiers and the accuracy is then calculated by comparing the predicted y values with the true y_test values. (Accuracy for the decision tree varies, for constant values, add random_state).

Reporting of Task 2B

Firstly, the training and testing x and y values which were imputed and standardised from 2a, is loaded by reading four csv files generated in task 2a. This is to reduce repeated code and since K-means is distance based algorithms it will be affected by the scaling. The next step generates features using interaction term pairs by using PolynomialFeatures with interaction_only=True, this generates features using original features with values a and b to form a new feature with value a*b. To create the feature from k-means clustering, the standardised, cleansed and imputed world.csv is used (from 2a) to generate coordinates with all combinations from the features for K-means clustering. To find the optimal value for k, the elbow method is used with k ranging from 1 to 15. This is because K-means aims to choose centroids that minimise the inertia (the sum of squares to the closest cluster center). By plotting a graph showing the inertia we can pick the point after which the inertia decreases linearly, this turns out to be k = 4. Once k is chosen, we can train K-means and plot the cluster centers in (red). The coordinates of the cluster centers, their corresponding number and the cluster number of the predicted test sets are printed to standard output.





Secondly, 4 features were picked (from 211) for first 5-NN classification, these were:

1. Domestic general government health expenditure per capita (current US$) (original feature)
2. Birth rate, crude (per 1,000 people) x
   Maternal mortality ratio (modeled estimate, per 100,000 live births) (interaction term pairs)
3. People using at least basic sanitation services (% of population) (original feature)
4. Access to electricity, rural (% of rural population) x
   Individuals using the Internet (% of population) (interaction term pairs)

These features were chosen as it has a strong correlation to life expectancy at birth for a country. The second and fourth features were products of two features, the features which they are made from are related which provides a more fine grain discriminant measure. Since these were standardised values, there are negatives we need to account for. For example, the higher the second feature, the more underdeveloped a country is (low y value). So if the standardised values were both negatives (both low - indicators of a developed country, high y), by multiplying them together it generates a positive number which will cause K-nn to misinterpret the data as low. Therefore if two of the numbers were negative, the negative sign is kept (for feature 2 and 4). K-means was not used as it did not account for double negatives so the data does not have a lot of value. And the cluster number does not tell us about which y value it corresponds to. These values from the 4 features were then put into 5-NN for training.

```
 - Output -
[ 2020-05-30 20:43:46.593616 ] Running program (submission/task2b.py).
[ 2020-05-30 20:43:46.597544 ] Program output stream:
Cluster center points:
 [[-0.66968418  0.86172329]
 [-0.58210321 -0.62739486]
 [ 1.00825056 -0.81638149]
 [ 1.18602053  1.13099409]]
Their corresponding cluster number:
 [0 1 2 3]
Predicting cluster numbers of testing set:
 [2 2 2 ... 3 2 2]
Accuracy of feature engineering: 73.770%
Accuracy of PCA: 75.410%
Accuracy of first four features: 75.410%
```

Thirdly, PCA was used to generate 4 principal components for the second 5-NN classification, these were new sets of features which were orthogonal to each other. The training data sets were then used to fit the 5-NN algorithm and accuracy calculated. Finally, the first four features of the dataset in world.csv was chosen to perform 5-NN classification.

The accuracy of using PCA and first four features were the same and highest out of the methods used. PCA is this high because all the original features are related to an extent with the y (life expectancy) values and PCA takes all the features into account to create 4 new features using a function, these linear combinations of features can capture more variability of data than feature engineering or plainly selecting the first four features. Plainly selecting the first four features yielded a high accuracy score might be because these four features might've strongly correlated well with the y values, giving more accurate results than interactive term pairs.

Feature engineering's accuracy was the lowest, this might be because the features were dependent on a class. A better method to select the best features is to use a chi-square test which tests the independence of pairs and it also takes into account the sample size which is good for understanding the value and its significance better. If the pair is not independent it is a good feature. Also, feature engineering is difficult to control as filtering features may remove most important details (which would've given a better accuracy score). Feature engineering also has potential issues where it is difficult to control the interdependence between features.

In conclusion, even though the accuracy for the first four features and for feature engineering was quite high, it is not reliable as dimension reduction should preserve the characteristics of the data, not remove potentially important features. If a pair of features were close to each other, after dimensionality reduction it should still be. Feature engineering is not that reliable as the dimensionality reduction was too simple, even if we generated 190 more features (linked each by 2 original features), only 4 were used. The characteristics of the data were not preserved. In this case we should perform feature engineering in a different way using a more complex function to make use of more features rather than just 2. Out of the 3 feature generation methods, PCA remains the most reliable as it reduces dimensionality by taking account of all the features while preserving the characteristics.